Guidelines Overview

William L. Fithen, Software Engineering Institute [vita[3]]

2005-10-03

All systems have vulnerabilities, either in the technology from which they are constructed or in the behaviors of the people who use them.

Introduction

The Build Security In Guidelines is a taxonomy of mid-level engineering concerns that were derived from the vulnerability database accumulated by the CERT® Coordination Center over its 15-year history. In general, these concerns are less abstract than the Build Security In Principles[5]—which are intended to be enduring top-level concerns—and more abstract than the Build Security In Coding Rules[6]—which are intended to be precise, specific implementation advice.

The Taxonomy

1. Assume that Human Behavior Will Introduce Vulnerabilities into Your System[8]
2. Assume that Technology Will Contain Vulnerabilities
    1. Follow the Rules Regarding Concurrency Management[9]
    2. Design Configuration Subsystems Correctly and Distribute Safe Default Configurations[10]
    3. Carefully Study Other Systems Before Incorporating Them into Your System Through Delegation[11]
    4. If Emulation of Another System Is Necessary, Ensure that It Is as Correct and Complete as Possible[12]
    5. Handle All Errors Safely[13]
    6. Validate All Input as Precisely as Possible
        1. Ensure that Input Is Properly Canonicalized[14]
        2. Be Suspicious about Trusting Unauthenticated External Representation of Internal Data Structures[15]
        3. Do Not Use the "%n" Format String Specifier[16]
        4. Treat the Entire Inherited Process Context as Unvalidated Input[17]
        5. Never Use Unvalidated Input as Part of a Directive to any Internal Component[18]

3. daisy:320 (Fithen, William L.)
5. daisy:79 (Principles)
6. daisy:76 (Coding Rules)
8. daisy:344 (Assume that Human Behavior Will Introduce Vulnerabilities into Your System)
9. daisy:332 (Follow the Rules Regarding Concurrency Management)
10. daisy:333 (Design Configuration Subsystems Correctly and Distribute Safe Default Configurations)
11. daisy:336 (Carefully Study Other Systems Before Incorporating Them into Your System)
12. daisy:337 (If Emulation of Another System Is Necessary, Ensure that It Is as Correct and Complete as Possible)
13. daisy:338 (Handle All Errors Safely)
14. daisy:331 (Ensure that Input Is Properly Canonicalized)
15. daisy:339 (Be Suspicious about Trusting Unauthenticated External Representation of Internal Data Structures)
16. daisy:340 (Do Not Use the "%n" Format String Specifier)
17. daisy:341 (Treat the Entire Inherited Process Context as Unvalidated Input)
18. daisy:342 (Never Use Unvalidated Input as Part of a Directive to any Internal Component)

7. Use All Security Mechanisms Correctly
    1. Use Authentication Mechanisms, Where Appropriate, Correctly[19]
    2. Use Authorization Mechanisms Correctly[20]
    3. Use Well-Known Cryptography Appropriately and Correctly[21]
8. Do Not Allow Your System to Ever Use or Depend on Language Behaviors that Are "Undefined"
    1. Ensure that the Bounds of No Memory Region Are Violated[22]
    2. Clear Discarded Storage that Contained Secrets and Do Not Read Uninitialized Storage[23]
    3. Do Not Perform Arithmetic with Unvalidated Input[24]

Description

In every phase of a system's development, under particular conditions, features added—or omitted—can introduce security vulnerabilities. To produce a safe and secure system, the competent, security-conscious engineer must

- learn the meaning of software assurance and be knowledgeable in the practice of supporting techniques,
- recognize the security implications of all functional requirements,
- recognize the security implications of missing requirements,
- recognize emergent behaviors in the system that have security implications,
- recognize the implications of an evolving deployment environment on the system,
- translate those implications into additional system requirements,
- design features to meet those requirements,
- recognize the security implications of the included and omitted features,
- add, modify, or remove features accordingly,
- recognize the security implications of the system's implementation,
- correct any defects in the implementation,
- understand how to test the system for compliance with security requirements, and
- be able to use software assurance techniques to demonstrate the assurance attributes of the system.

A failure in any of these, and more, can leave the system with security vulnerabilities.

19. daisy:321 (Use Authentication Mechanisms, Where Appropriate, Correctly)
20. daisy:322 (Use Authorization Mechanisms Correctly)
21. daisy:334 (Use Well-Known Cryptography Appropriately and Correctly)
22. daisy:328 (Ensure that the Bounds of No Memory Region Are Violated)
23. daisy:335 (Clear Discarded Storage that Contained Secrets and Do Not Read Uninitialized Storage)
24. daisy:343 (Do Not Perform Arithmetic with Unvalidated Input)

Fields

Name	Value
Copyright Holder	SEI

Fields

Name	Value
is-content-area-overview	true
Content Areas	Knowledge/Guidelines
SDLC Relevance	Implementation
Workflow State	Publishable

1. http://www.sei.cmu.edu/about/legal-permissions.html